Chapter 8. Setting up a WireGuard VPN

WireGuard is a high-performance VPN solution that runs in the Linux kernel. It uses modern cryptography and is easier to configure than many other VPN solutions. Additionally, WireGuard's small codebase reduces the surface for attacks and, therefore, improves security. For authentication and encryption, WireGuard uses keys similar to SSH.

> **Important**
>
> WireGuard is provided as a Technology Preview only. Technology Preview features are not supported with Red Hat production Service Level Agreements (SLAs), might not be functionally complete, and Red Hat does not recommend using them for production. These previews provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.
>
> See Technology Preview Features Support Scope on the Red Hat Customer Portal for information about the support scope for Technology Preview features.

To set up a WireGuard VPN, you must complete the following steps. You can perform most steps by using different options:

1. Create public and private keys for every host in the VPN.
2. Configure the WireGuard server by using nmcli, nmtui, nm-connection-editor, or the wg-quick service.
3. Configure firewalld on the WireGuard server by using the command line or graphical interface.
4. Configure the WireGuard client by using nmcli, nm-connection-editor, or the wg-quick service.

WireGuard operates on the network layer (layer 3). Therefore, you cannot use DHCP and must assign static IP addresses or IPv6 link-local addresses to the tunnel devices on both the server and clients.

> **Important**
>
> You can use WireGuard only if the Federal Information Processing Standard (FIPS) mode in RHEL is disabled.

Note that all hosts that participate in a WireGuard VPN are peers. This documentation uses the terms `client` to describe hosts that establish a connection and `server` to describe the host with the fixed hostname or IP address that the clients connect to and optionally route all traffic through this server.

8.1. Protocols and primitives used by WireGuard

WireGuard uses the following protocols and primitives:

- ChaCha20 for symmetric encryption, authenticated with Poly1305, using Authenticated Encryption with Associated Data (AEAD) construction as described in RFC7539
- Curve25519 for Elliptic-curve Diffie–Hellman (ECDH) key exchange
- BLAKE2s for hashing and keyed hashing, as described in RFC7693
- SipHash24 for hash table keys
- HKDF for key derivation, as described in RFC5869

8.2. How WireGuard uses tunnel IP addresses, public keys,

and remote endpoints

When WireGuard sends a network packet to a peer:

1. WireGuard reads the destination IP from the packet and compares it to the list of allowed IP addresses in the local configuration. If the peer is not found, WireGuard drops the packet.
2. If the peer is valid, WireGuard encrypts the packet using the peer's public key.
3. The sending host looks up the most recent Internet IP address of the host and sends the encrypted packet to it.

When WireGuard receives a packet:

1. WireGuard decrypts the packet using private key of the remote host.
2. WireGuard reads the internal source address from the packet and looks up whether the IP is configured in the list of allowed IP addresses in the settings for the peer on the local host. If the source IP is on the allowlist, WireGuard accepts the packet. If the IP address is not on the list, WireGuard drops the packet.

The association of public keys and allowed IP addresses is called `Cryptokey Routing Table` . This means that the list of IP addresses behaves similar to a routing table when sending packets, and as a kind of access control list when receiving packets.

8.3. Using a WireGuard client behind NAT and firewalls

WireGuard uses the UDP protocol and transmits data only when a peer sends packets. Stateful firewalls and network address translation (NAT) on routers track connections to enable a peer behind NAT or a firewall to receive packets.

To keep the connection active, WireGuard supports `persistent keepalives`. This means you can set an interval at which WireGuard sends keepalive packets. By default, the persistent keep-alive feature is disabled to reduce network traffic. Enable this feature on the client if you use the client in a network with NAT or if a firewall closes the connection after some time of inactivity.

8.4. Creating private and public keys to be used in WireGuard connections

WireGuard uses base64-encoded private and public keys to authenticate hosts to each other. Therefore, you must create the keys on each host that participates in the WireGuard VPN.

> **Important**
>
> For secure connections, create different keys for each host, and ensure that you only share the public key with the remote WireGuard host. Do not use the example keys used in this documentation.

Procedure

1. Install the `wireguard-tools` package:

```
# dnf install wireguard-tools
```

2. Create a private key and a corresponding public key for the host:

```
# wg genkey | tee /etc/wireguard/$HOSTNAME.private.key | wg pubkey > /etc/wireguard/$HOSTNAME.public.key
```

You will need the content of the key files, but not the files themselves. However, Red Hat recommends keeping the files in case that you need to remember the keys in future.

3. Set secure permissions on the key files:

```
# chmod 600 /etc/wireguard/$HOSTNAME.private.key
/etc/wireguard/$HOSTNAME.public.key
```

4. Display the private key:

```
# cat /etc/wireguard/$HOSTNAME.private.key
YFAnE0psgIdiAF7XR4abxiwVRnlMfeltxu10s/c4JXg=
```

You will need the private key to configure the WireGuard connection on the local host. Do not share the private key.

5. Display the public key:

```
# cat /etc/wireguard/$HOSTNAME.public.key
UtjqCJ57DeAscYKRfp7cFGiQqdONRn69u249Fa4O6BE=
```

You will need the public key to configure the WireGuard connection on the remote host.

ADDITIONAL RESOURCES

The `wg(8)` man page

8.5. Configuring a WireGuard server by using nmcli

You can configure the WireGuard server by creating a connection profile in NetworkManager. Use this method to let NetworkManager manage the WireGuard connection.

This procedure assumes the following settings:

- Server:
  - Private key: `YFAnE0psgIdiAF7XR4abxiwVRnlMfeltxu10s/c4JXg=`
  - Tunnel IPv4 address: `192.0.2.1/24`
  - Tunnel IPv6 address: `2001:db8:1::1/32`
- Client:
  - Public key: `bnwfQcC8/g2i4vvEqcRUM2e6Hi3Nskk6G9t4r26nFVM=`
  - Tunnel IPv4 address: `192.0.2.2/24`
  - Tunnel IPv6 address: `2001:db8:1::2/32`

Prerequisites

- You have generated the public and private key for both the server and client.
- You know the following information:
  - The private key of the server
  - The static tunnel IP addresses and subnet masks of the client
  - The public key of the client
  - The static tunnel IP addresses and subnet masks of the server

Procedure

1. Add a NetworkManager WireGuard connection profile:

```
# nmcli connection add type wireguard con-name server-wg0 ifname wg0
autoconnect no
```

This command creates a profile named `server-wg0` and assigns the virtual interface `wg0` to it. To prevent the connection from starting automatically after you add it without finalizing the configuration, disable the `autoconnect` parameter.

2. Set the tunnel IPv4 address and subnet mask of the server:

```
# nmcli connection modify server-wg0 ipv4.method manual
ipv4.addresses 192.0.2.1/24
```

3. Set the tunnel IPv6 address and subnet mask of the server:

```
# nmcli connection modify server-wg0 ipv6.method manual
ipv6.addresses 2001:db8:1::1/32
```

4. Add the server's private key to the connection profile:

```
# nmcli connection modify server-wg0 wireguard.private-key
"YFAnE0psgIdiAF7XR4abxiwVRnlMfeltxu10s/c4JXg="
```

5. Set the port for incoming WireGuard connections:

```
# nmcli connection modify server-wg0 wireguard.listen-port 51820
```

Always set a fixed port number on hosts that receive incoming WireGuard connections. If you do not set a port, WireGuard uses a random free port each time you activate the `wg0` interface.

6. Add peer configurations for each client that you want to allow to communicate with this server. You must add these settings manually, because the `nmcli` utility does not support setting the corresponding connection properties.

   i. Edit the `/etc/NetworkManager/system-connections/server-wg0.nmconnection` file, and append:

```
[wireguard-peer.bnwfQcC8/g2i4vvEqcRUM2e6Hi3Nskk6G9t4r26nFVM=]
allowed-ips=192.0.2.2;2001:db8:1::2;
```

- The `[wireguard-peer.<public_key_of_the_client>]` entry defines the peer section of the client, and the section name contains the public key of the client.
- The `allowed-ips` parameter sets the tunnel IP addresses of the client that are allowed to send data to this server.

  Add a section for each client.

ii. Reload the `server-wg0` connection profile:

```
# nmcli connection load /etc/NetworkManager/system-connections/server-wg0.nmconnection
```

7. Optional: Configure the connection to start automatically, enter:

```
# nmcli connection modify server-wg0 autoconnect yes
```

8. Reactivate the `server-wg0` connection:

```
# nmcli connection up server-wg0
```

Next steps

- Configure the firewalld service on the WireGuard server.

Verification

1. Display the interface configuration of the `wg0` device:

```
# wg show wg0
interface: wg0
  public key: UtjqCJ57DeAscYKRfp7cFGiQqdONRn69u249Fa4O6BE=
  private key: (hidden)
  listening port: 51820

peer: bnwfQcC8/g2i4vvEqcRUM2e6Hi3Nskk6G9t4r26nFVM=
  allowed ips: 192.0.2.2/32, 2001:db8:1::2/128
```

To display the private key in the output, use the `WG_HIDE_KEYS=never wg show wg0` command.

2. Display the IP configuration of the `wg0` device:

```
# ip address show wg0
20: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue
state UNKNOWN group default qlen 1000
    link/none
    inet 192.0.2.1/24 brd 192.0.2.255 scope global noprefixroute wg0
       valid_lft forever preferred_lft forever
    inet6 2001:db8:1::1/32 scope global noprefixroute
       valid_lft forever preferred_lft forever
    inet6 fe80::3ef:8863:1ce2:844/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

ADDITIONAL RESOURCES

The `wg(8)` man page

The `WireGuard setting` section in the `nm-settings(5)` man page

8.6. Configuring a WireGuard server by using nmtui

You can configure the WireGuard server by creating a connection profile in NetworkManager. Use this method to let NetworkManager manage the WireGuard connection.

This procedure assumes the following settings:

- Server:
  - Private key: `YFAnE0psgIdiAF7XR4abxiwVRnlMfeltxu10s/c4JXg=`
  - Tunnel IPv4 address: `192.0.2.1/24`
  - Tunnel IPv6 address: `2001:db8:1::1/32`
- Client:
  - Public key: `bnwfQcC8/g2i4vvEqcRUM2e6Hi3Nskk6G9t4r26nFVM=`
  - Tunnel IPv4 address: `192.0.2.2/24`
  - Tunnel IPv6 address: `2001:db8:1::2/32`

Prerequisites

- You have generated the public and private key for both the server and client.
- You know the following information:
  - The private key of the server
  - The static tunnel IP addresses and subnet masks of the client
  - The public key of the client
  - The static tunnel IP addresses and subnet masks of the server
- You installed the `NetworkManager-tui` package.

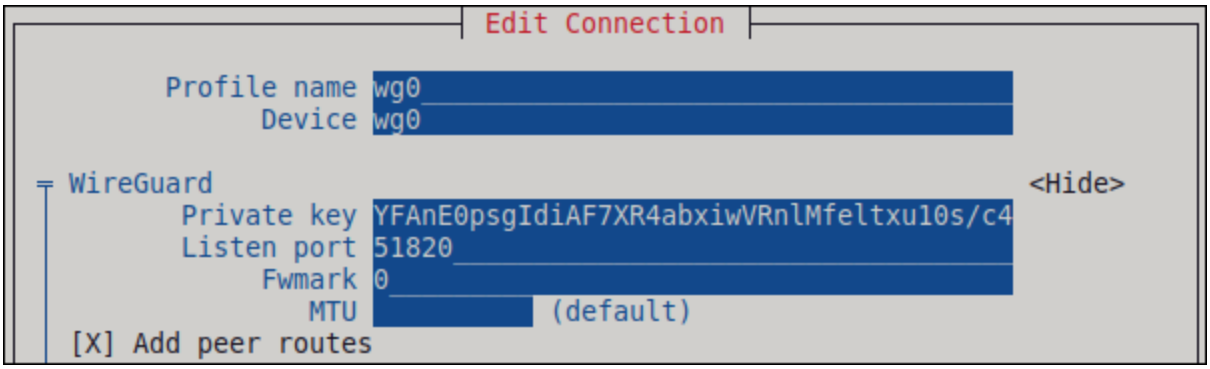
Procedure

1. Start the `nmtui` application:

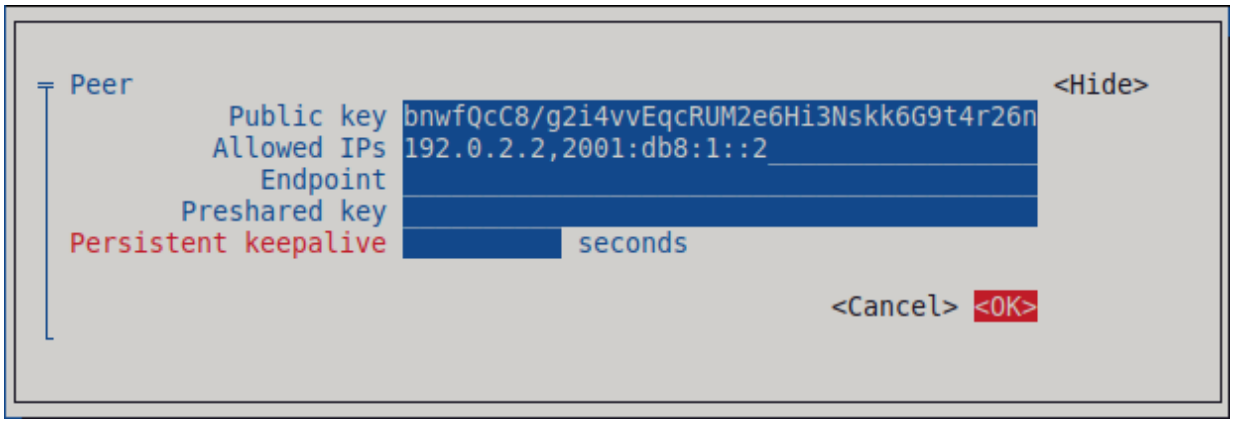
```
# nmtui
```

2. Select `Edit a connection`, and press `Enter`.
3. Select **[Add]**, and press `Enter`.
4. Select the `WireGuard` connection type in the list, and press `Enter`.
5. In the `Edit connection` window:
   i. Enter the name of the connection and the virtual interface, such as `wg0`, that NetworkManager should assign to the connection.
   ii. Enter the private key of the server.
   iii. Set the listen port number, such as `51820`, for incoming WireGuard connections.

   Always set a fixed port number on hosts that receive incoming WireGuard connections. If you do not set a port, WireGuard uses a random free port each time you activate the interface.



   iv. Click **[Add]** next to the `Peers` pane:
      i. Enter the public key of the client.
      ii. Set the `Allowed IPs` field to the tunnel IP addresses of the client that are allowed to send data to this server.
      iii. Select **[OK]**, and press `Enter`.



v. Select **[Show]** next to `IPv4 Configuration`, and press `Enter`.

   i. Select the IPv4 configuration method `Manual`.

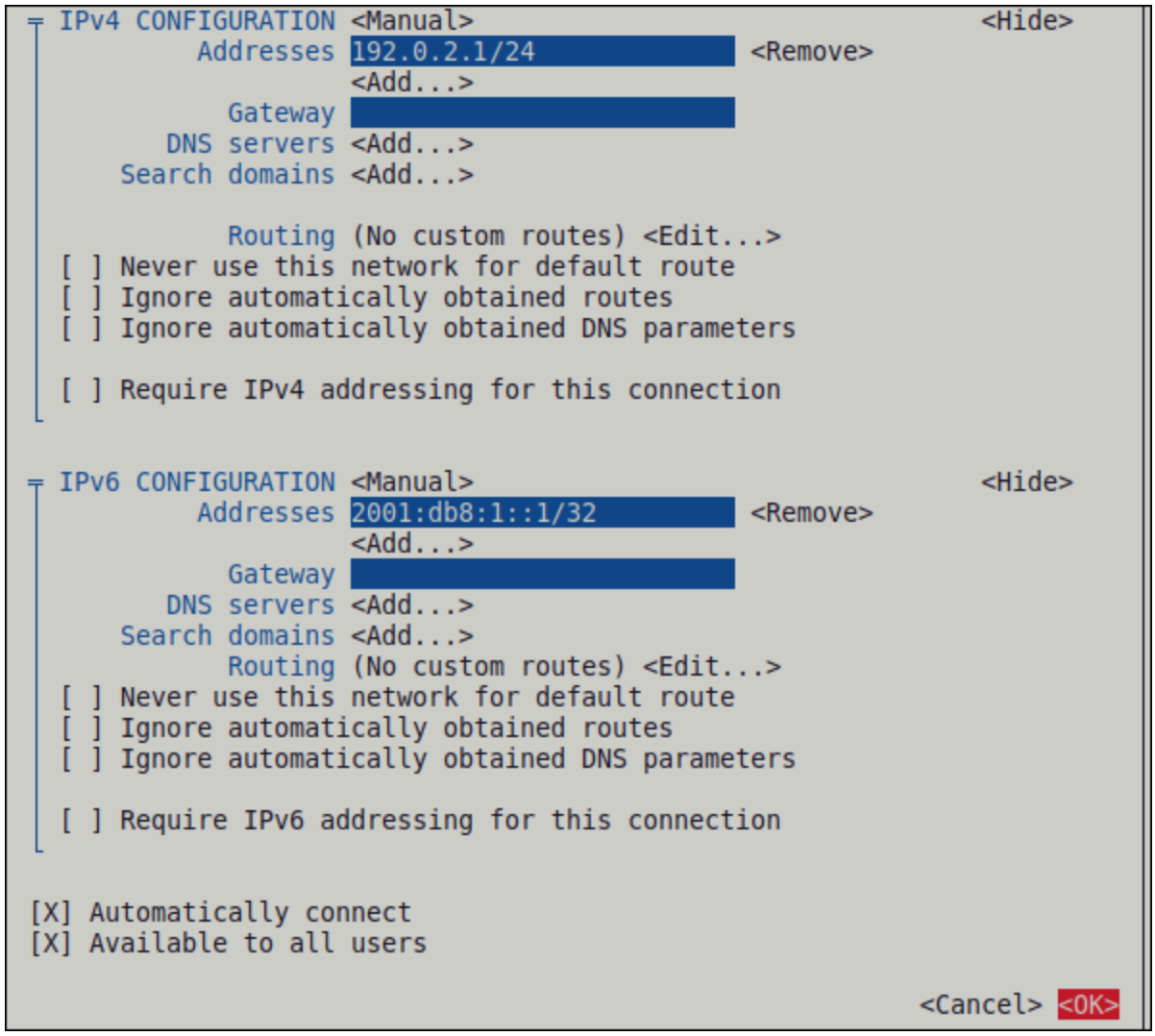
   ii. Enter the tunnel IPv4 address and the subnet mask. Leave the `Gateway` field empty.

vi. Select **[Show]** next to `IPv6 Configuration`, and press `Enter`.

   i. Select the IPv6 configuration method `Manual`.

   ii. Enter the tunnel IPv6 address and the subnet mask. Leave the `Gateway` field empty.

vii. Select **[OK]**, and press `Enter`



6. In the window with the list of connections, select **[Back]**, and press `Enter`.
7. In the `NetworkManager TUI` main window, select **[Quit]**, and press `Enter`.

Next steps

- Configure the firewalld service on the WireGuard server.

Verification

1. Display the interface configuration of the `wg0` device:

```
# wg show wg0
interface: wg0
  public key: UtjqCJ57DeAscYKRfp7cFGiQqdONRn69u249Fa4O6BE=
  private key: (hidden)
  listening port: 51820

peer: bnwfQcC8/g2i4vvEqcRUM2e6Hi3Nskk6G9t4r26nFVM=
  allowed ips: 192.0.2.2/32, 2001:db8:1::2/128
```

To display the private key in the output, use the `WG_HIDE_KEYS=never wg show wg0` command.

2. Display the IP configuration of the `wg0` device:

```
# ip address show wg0
20: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue
state UNKNOWN group default qlen 1000
    link/none
    inet 192.0.2.1/24 brd 192.0.2.255 scope global noprefixroute wg0
       valid_lft forever preferred_lft forever
    inet6 2001:db8:1::1/32 scope global noprefixroute
       valid_lft forever preferred_lft forever
    inet6 fe80::3ef:8863:1ce2:844/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

ADDITIONAL RESOURCES

The `wg(8)` man page

8.7. Configuring a WireGuard server by using nm-connection-editor

You can configure the WireGuard server by creating a connection profile in NetworkManager. Use this method to let NetworkManager manage the WireGuard connection.

Prerequisites

- You have generated the public and private key for both the server and client.
- You know the following information:
  - The private key of the server
  - The static tunnel IP addresses and subnet masks of the client
  - The public key of the client
  - The static tunnel IP addresses and subnet masks of the server

Procedure

1. Open a terminal, and enter:

```
# nm-connection-editor
```

2. Add a new connection by clicking the **[+]** button.
3. Select the `WireGuard` connection type, and click **[Create]**.
4. Optional: Update the connection name.
5. On the `General` tab, select `Connect automatically with priority`. Optionally, set a priority value.
6. On the `WireGuard` tab:
   i. Enter the name of the virtual interface, such as `wg0`, that NetworkManager should assign to the connection.
   ii. Enter the private key of the server.

iii. Set the listen port number, such as `51820`, for incoming WireGuard connections.

Always set a fixed port number on hosts that receive incoming WireGuard connections. If you do not set a port, WireGuard uses a random free port each time you activate the interface.

iv. Click **[Add]** to add peers:

i. Enter the public key of the client.

ii. Set the `Allowed IPs` field to the tunnel IP addresses of the client that are allowed to send data to this server.

iii. Click **[Apply]**.

7. On the `IPv4 Settings` tab:

i. Select `Manual` in the `Method` list.

ii. Click **[Add]** to enter the tunnel IPv4 address and the subnet mask. Leave the `Gateway` field empty.

8. On the `IPv6 Settings` tab:

i. Select `Manual` in the `Method` list.

ii. Click **[Add]** to enter the tunnel IPv6 address and the subnet mask. Leave the `Gateway` field empty.

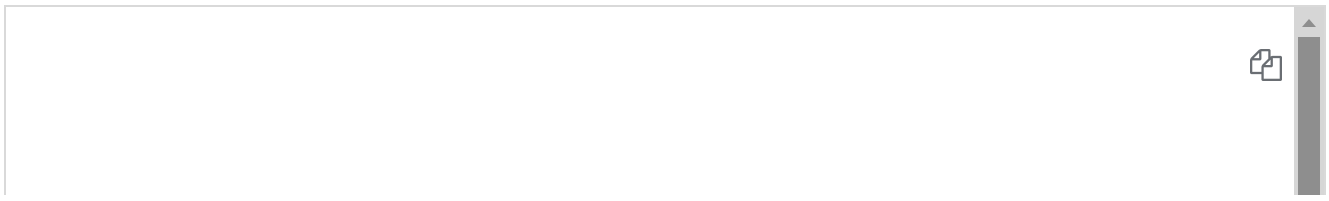
9. Click **[Save]** to store the connection profile.

Next steps

- Configure the firewalld service on the WireGuard server.

Verification

1. Display the interface configuration of the `wg0` device:



```
# wg show wg0
interface: wg0
  public key: UtjqCJ57DeAscYKRfp7cFGiQqdONRn69u249Fa4O6BE=
  private key: (hidden)
  listening port: 51820

peer: bnwfQcC8/g2i4vvEqcRUM2e6Hi3Nskk6G9t4r26nFVM=
  allowed ips: 192.0.2.2/32, 2001:db8:1::2/128
```

To display the private key in the output, use the `WG_HIDE_KEYS=never wg show wg0` command.

2. Display the IP configuration of the `wg0` device:

```
# ip address show wg0
20: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue
state UNKNOWN group default qlen 1000
    link/none
    inet 192.0.2.1/24 brd 192.0.2.255 scope global noprefixroute wg0
       valid_lft forever preferred_lft forever
    inet6 2001:db8:1::1/32 scope global noprefixroute
       valid_lft forever preferred_lft forever
    inet6 fe80::3ef:8863:1ce2:844/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

ADDITIONAL RESOURCES

The `wg(8)` man page

8.8. Configuring a WireGuard server by using the wg-quick service

You can configure the WireGuard server by creating a configuration file in the `/etc/wireguard/` directory. Use this method to configure the service independently from NetworkManager.

This procedure assumes the following settings:

- Server:
  - Private key: `YFAnE0psgIdiAF7XR4abxiwVRnlMfeltxu10s/c4JXg=`
  - Tunnel IPv4 address: `192.0.2.1/24`
  - Tunnel IPv6 address: `2001:db8:1::1/32`
- Client:
  - Public key: `bnwfQcC8/g2i4vvEqcRUM2e6Hi3Nskk6G9t4r26nFVM=`
  - Tunnel IPv4 address: `192.0.2.2/24`
  - Tunnel IPv6 address: `2001:db8:1::2/32`

Prerequisites

- You have generated the public and private key for both the server and client.
- You know the following information:
  - The private key of the server
  - The static tunnel IP addresses and subnet masks of the client
  - The public key of the client
  - The static tunnel IP addresses and subnet masks of the server

Procedure

1. Install the `wireguard-tools` package:

```
# dnf install wireguard-tools
```

2. Create the `/etc/wireguard/wg0.conf` file with the following content:

```
[Interface]
Address = 192.0.2.1/24, 2001:db8:1::1/32
ListenPort = 51820
PrivateKey = YFAnE0psgIdiAF7XR4abxiwVRnlMfeltxu10s/c4JXg=

[Peer]
PublicKey = bnwfQcC8/g2i4vvEqcRUM2e6Hi3Nskk6G9t4r26nFVM=
AllowedIPs = 192.0.2.2, 2001:db8:1::2
```

- The `[Interface]` section describes the WireGuard settings of the interface on the server:
  - `Address` : A comma-separated list of the server's tunnel IP addresses.
  - `PrivateKey` : The private key of the server.
  - `ListenPort` : The port on which WireGuard listens for incoming UDP connections.

    Always set a fixed port number on hosts that receive incoming WireGuard connections. If you do not set a port, WireGuard uses a random free port each time you activate the `wg0` interface.
- Each `[Peer]` section describes the settings of one client:
  - `PublicKey` : The public key of the client.
  - `AllowedIPs` : The tunnel IP addresses of the client that are allowed to send data to this server.

3. Enable and start the WireGuard connection:

```
# systemctl enable --now wg-quick@wg0
```

The systemd instance name must match the name of the configuration file in the `/etc/wireguard/` directory without the `.conf` suffix. The service also uses this name for the virtual network interface.

Next steps

- Configure the firewalld service on the WireGuard server.

Verification

1. Display the interface configuration of the `wg0` device:

```
# wg show wg0
interface: wg0
  public key: UtjqCJ57DeAscYKRfp7cFGiQqdONRn69u249Fa4O6BE=
  private key: (hidden)
  listening port: 51820

peer: bnwfQcC8/g2i4vvEqcRUM2e6Hi3Nskk6G9t4r26nFVM=
  allowed ips: 192.0.2.2/32, 2001:db8:1::2/128
```

   To display the private key in the output, use the `WG_HIDE_KEYS=never wg show wg0` command.

2. Display the IP configuration of the `wg0` device:

```
# ip address show wg0
20: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue
state UNKNOWN group default qlen 1000
    link/none
    inet 192.0.2.1/24 scope global wg0
       valid_lft forever preferred_lft forever
    inet6 2001:db8:1::1/32 scope global
       valid_lft forever preferred_lft forever
```

ADDITIONAL RESOURCES

The `wg(8)` man page

The `wg-quick(8)` man page

8.9. Configuring firewalld on a WireGuard server by using the command line

You must configure the `firewalld` service on the WireGuard server to allow incoming connections from clients. Additionally, if clients should be able to use the WireGuard server as the default gateway and route all traffic through the tunnel, you must enable masquerading.

Procedure

1. Open the WireGuard port for incoming connections in the `firewalld` service:

```
# firewall-cmd --permanent --add-port=51820/udp --zone=public
```

2. If clients should route all traffic through the tunnel and use the WireGuard server as the default gateway, enable masquerading for the `public` zone:

```
# firewall-cmd --permanent --zone=public --add-masquerade
```

3. Reload the `firewalld` rules.

```
# firewall-cmd --reload
```

Verification

- Display the configuration of the `public` zone:

```
# firewall-cmd --list-all
public (active)
  ...
  ports: 51820/udp
  masquerade: yes
  ...
```

ADDITIONAL RESOURCES

The `firewall-cmd(1)` man page

8.10. Configuring firewalld on a WireGuard server by using the graphical interface

You must configure the `firewalld` service on the WireGuard server to allow incoming connections from clients. Additionally, if clients should be able to use the WireGuard server as the default gateway and route all traffic through the tunnel, you must enable masquerading.

Procedure

1. Press the `Super` key, enter `firewall` , and select the `Firewall` application from the results.
2. Select `Permanent` in the `Configuration` list.
3. Select the `public` zone.
4. Allow incoming connections to the WireGuard port:

i. On the `Ports` tab, click **[Add]**.

ii. Enter the port number you set for incoming WireGuard connections:

iii. Select `udp` from the `Protocol` list.

iv. Click **[OK]**.

5. If clients should route all traffic through the tunnel and use the WireGuard server as the default gateway:

i. Navigate to the `Masquerading` tab of the `public` zone.

ii. Select `Masquerade zone`.

6. Select **Options** → **Reload Firewalld**.

Verification

- Display the configuration of the `public` zone:

```
# firewall-cmd --list-all
public (active)
  ...
  ports: 51820/udp
  masquerade: yes
  ...
```

8.11. Configuring a WireGuard client by using nmcli

You can configure a WireGuard client by creating a connection profile in NetworkManager. Use this method to let NetworkManager manage the WireGuard connection.

This procedure assumes the following settings:

- Client:

- Private key: `aPUcp5vHz8yMLrzk8SsDyYnV33IhE/k20e52iKJFV0A=`
- Tunnel IPv4 address: `192.0.2.2/24`
- Tunnel IPv6 address: `2001:db8:1::2/32`

- Server:
  - Public key: `UtjqCJ57DeAscYKRfp7cFGiQqdONRn69u249Fa4O6BE=`
  - Tunnel IPv4 address: `192.0.2.1/24`
  - Tunnel IPv6 address: `2001:db8:1::1/32`

Prerequisites

- You have generated the public and private key for both the server and client.
- You know the following information:
  - The private key of the client
  - The static tunnel IP addresses and subnet masks of the client
  - The public key of the server
  - The static tunnel IP addresses and subnet masks of the server

Procedure

1. Add a NetworkManager WireGuard connection profile:

```
# nmcli connection add type wireguard con-name client-wg0 ifname wg0 autoconnect no
```

   This command creates a profile named `client-wg0` and assigns the virtual interface `wg0` to it. To prevent the connection from starting automatically after you add it without finalizing the configuration, disable the `autoconnect` parameter.

2. Optional: Configure NetworkManager so that it does not automatically start the `client-wg` connection:

```
# nmcli connection modify client-wg0 autoconnect no
```

3. Set the tunnel IPv4 address and subnet mask of the client:

```
# nmcli connection modify client-wg0 ipv4.method manual
ipv4.addresses 192.0.2.2/24
```

4. Set the tunnel IPv6 address and subnet mask of the client:

```
# nmcli connection modify client-wg0 ipv6.method manual
ipv6.addresses 2001:db8:1::2/32
```

5. If you want to route all traffic through the tunnel, set the tunnel IP addresses of the server as the default gateway:

```
# nmcli connection modify client-wg0 ipv4.gateway 192.0.2.1
ipv6.gateway 2001:db8:1::1
```

   Routing all traffic through the tunnel requires that you set, in a later step, the `allowed-ips` on the this client to `0.0.0.0/0;::/0` .

   Note that routing all traffic through the tunnel can impact the connectivity to other hosts based on the server routing and firewall configuration.

6. Add the client's private key to the connection profile:

```
# nmcli connection modify client-wg0 wireguard.private-key
"aPUcp5vHz8yMLrzk8SsDyYnV33IhE/k20e52iKJFV0A="
```

7. Add peer configurations for each server that you want to allow to communicate with this client. You must add these settings manually, because the `nmcli` utility does not support setting the corresponding connection properties.

   i. Edit the `/etc/NetworkManager/system-connections/client-wg0.nmconnection` file, and append:

```
[wireguard-peer.UtjqCJ57DeAscYKRfp7cFGiQqd0NRn69u249Fa406BE=]
endpoint=server.example.com:51820
allowed-ips=192.0.2.1;2001:db8:1::1;
persistent-keepalive=20
```

- The `[wireguard-peer.<public_key_of_the_server>]` entry defines the peer section of the server, and the section name has the public key of the server.
- The `endpoint` parameter sets the hostname or IP address and the port of the server. The client uses this information to establish the connection.
- The `allowed-ips` parameter sets a list of IP addresses that can send data to this client. For example, set the parameter to:
  - The tunnel IP addresses of the server to allow only the server to communicate with this client. The value in the example above configures this scenario.
  - `0.0.0.0/0;::/0;` to allow any remote IPv4 and IPv6 address to communicate with this client. Use this setting to route all traffic through the tunnel and use the WireGuard server as default gateway.
- The optional `persistent-keepalive` parameter defines an interval in seconds in which WireGuard sends a keep alive packet to the server. Set this parameter if you use the client in a network with network address translation (NAT) or if a firewall closes the UDP connection after some time of inactivity.

ii. Reload the `client-wg0` connection profile:

```
# nmcli connection load /etc/NetworkManager/system-connections/client-wg0.nmconnection
```

8. Reactivate the `client-wg0` connection:

```
# nmcli connection up client-wg0
```

Verification

1. Ping the IP addresses of the server:

```
# ping 192.0.2.1
# ping6 2001:db8:1::1
```

2. Display the interface configuration of the `wg0` device:

```
# wg show wg0
interface: wg0
  public key: bnwfQcC8/g2i4vvEqcRUM2e6Hi3Nskk6G9t4r26nFVM=
  private key: (hidden)
  listening port: 51820

peer: UtjqCJ57DeAscYKRfp7cFGiQqdONRn69u249Fa4O6BE=
  endpoint: server.example.com:51820
  allowed ips: 192.0.2.1/32, 2001:db8:1::1/128
  latest handshake: 1 minute, 41 seconds ago
  transfer: 824 B received, 1.01 KiB sent
  persistent keepalive: every 20 seconds
```

To display the private key in the output, use the `WG_HIDE_KEYS=never wg show wg0` command.

Note that the output has only the `latest handshake` and `transfer` entries if you have already sent traffic through the VPN tunnel.

3. Display the IP configuration of the `wg0` device:

```
# ip address show wg0
10: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue
state UNKNOWN group default qlen 1000
    link/none
    inet 192.0.2.2/24 brd 192.0.2.255 scope global noprefixroute wg0
       valid_lft forever preferred_lft forever
    inet6 2001:db8:1::2/32 scope global noprefixroute
       valid_lft forever preferred_lft forever
    inet6 fe80::73d9:6f51:ea6f:863e/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

ADDITIONAL RESOURCES

The `wg(8)` man page

The `WireGuard setting` section in the `nm-settings(5)` man page

8.12. Configuring a WireGuard client by using nmtui

You can configure a WireGuard client by creating a connection profile in NetworkManager. Use this method to let NetworkManager manage the WireGuard connection.

This procedure assumes the following settings:

- Client:
  - Private key: `aPUcp5vHz8yMLrzk8SsDyYnV33IhE/k20e52iKJFV0A=`
  - Tunnel IPv4 address: `192.0.2.2/24`
  - Tunnel IPv6 address: `2001:db8:1::2/32`
- Server:
  - Public key: `UtjqCJ57DeAscYKRfp7cFGiQqdONRn69u249Fa4O6BE=`
  - Tunnel IPv4 address: `192.0.2.1/24`
  - Tunnel IPv6 address: `2001:db8:1::1/32`

Prerequisites

- You have generated the public and private key for both the server and client.
- You know the following information:
  - The private key of the client
  - The static tunnel IP addresses and subnet masks of the client

- The public key of the server
- The static tunnel IP addresses and subnet masks of the server

- You installed the `NetworkManager-tui` package

Procedure

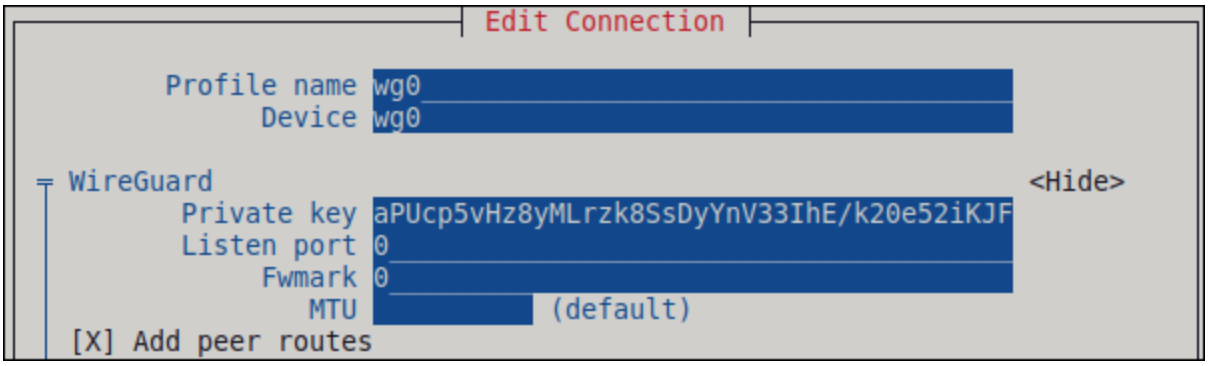
1. Start the `nmtui` application:

```
# nmtui
```

2. Select `Edit a connection`, and press `Enter`.
3. Select **[Add]**, and press `Enter`.
4. Select the `WireGuard` connection type in the list, and press `Enter`.
5. In the `Edit connection` window:

   i. Enter the name of the connection and the virtual interface, such as `wg0`, that NetworkManager should assign to the connection.

   ii. Enter the private key of the client.



   iii. Click **[Add]** next to the `Peers` pane:

      i. Enter the public key of the server.

      ii. Set the `Allowed IPs` field. For example, set it to:

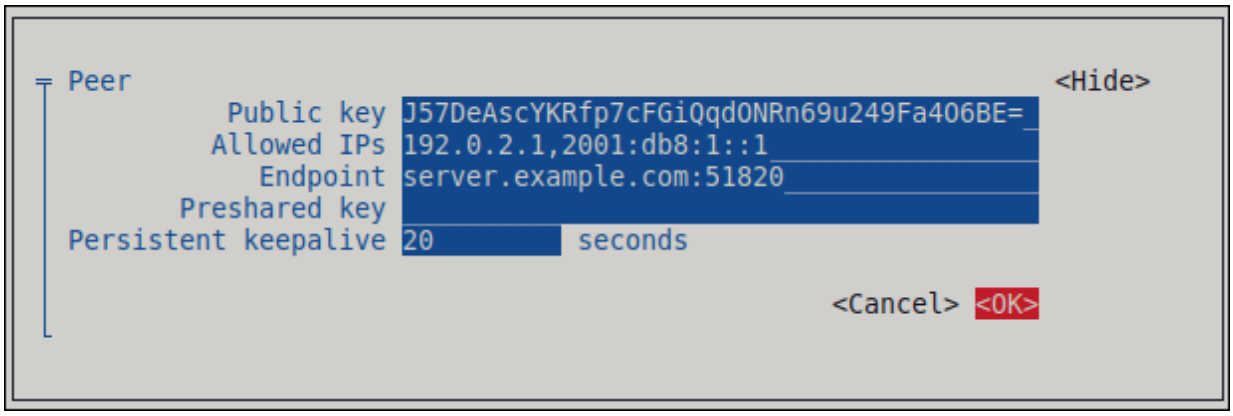
      - The tunnel IP addresses of the server to allow only the server to communicate with this client.

- `0.0.0.0/0,::/0` to allow any remote IPv4 and IPv6 address to communicate with this client. Use this setting to route all traffic through the tunnel and use the WireGuard server as default gateway.

iii. Enter the host name or IP address and port of the WireGuard server into the `Endpoint` field. Use the following format: *`hostname_or_IP:port_number`*

iv. Optional: If you use the client in a network with network address translation (NAT) or if a firewall closes the UDP connection after some time of inactivity, set a persistent keep alive interval in seconds. In this interval, the client sends a keepalive packet to the server.

v. Select **[OK]**, and press `Enter`.



iv. Select **[Show]** next to `IPv4 Configuration`, and press `Enter`.

i. Select the IPv4 configuration method `Manual`.

ii. Enter the tunnel IPv4 address and the subnet mask. Leave the `Gateway` field empty.

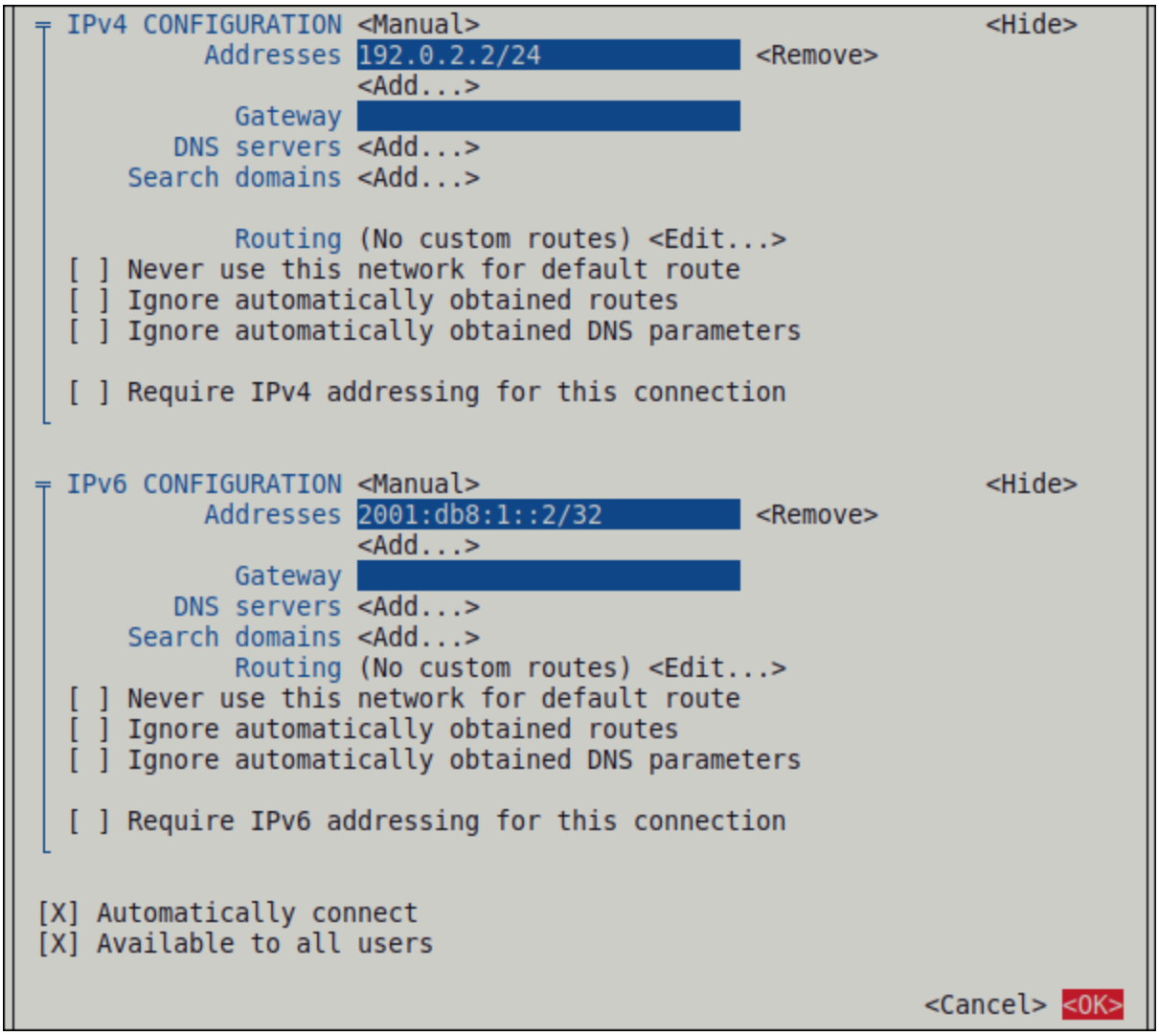
v. Select **[Show]** next to `IPv6 Configuration`, and press `Enter`.

i. Select the IPv6 configuration method `Manual`.

ii. Enter the tunnel IPv6 address and the subnet mask. Leave the `Gateway` field empty.

vi. Optional: Select `Automatically connect`.

vii. Select **[OK]**, and press `Enter`



6. In the window with the list of connections, select **[Back]**, and press `Enter`.
7. In the `NetworkManager TUI` main window, select **[Quit]**, and press `Enter`.

Verification

1. Ping the IP addresses of the server:

```
# ping 192.0.2.1
# ping6 2001:db8:1::1
```

2. Display the interface configuration of the `wg0` device:

```
# wg show wg0
interface: wg0
  public key: bnwfQcC8/g2i4vvEqcRUM2e6Hi3Nskk6G9t4r26nFVM=
  private key: (hidden)
  listening port: 51820

peer: UtjqCJ57DeAscYKRfp7cFGiQqdONRn69u249Fa4O6BE=
  endpoint: server.example.com:51820
  allowed ips: 192.0.2.1/32, 2001:db8:1::1/128
  latest handshake: 1 minute, 41 seconds ago
  transfer: 824 B received, 1.01 KiB sent
  persistent keepalive: every 20 seconds
```

To display the private key in the output, use the `WG_HIDE_KEYS=never wg show wg0` command.

Note that the output contains only the `latest handshake` and `transfer` entries i you have already sent traffic through the VPN tunnel.

3. Display the IP configuration of the `wg0` device:

```
# ip address show wg0
10: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue
state UNKNOWN group default qlen 1000
    link/none
    inet 192.0.2.2/24 brd 192.0.2.255 scope global noprefixroute wg0
       valid_lft forever preferred_lft forever
    inet6 2001:db8:1::2/32 scope global noprefixroute
       valid_lft forever preferred_lft forever
    inet6 fe80::73d9:6f51:ea6f:863e/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

ADDITIONAL RESOURCES

The `wg(8)` man page

8.13. Configuring a WireGuard client by using nm-connection-editor

You can configure a WireGuard client by creating a connection profile in NetworkManager. Use this method to let NetworkManager manage the WireGuard connection.

Prerequisites

- You have generated the public and private key for both the server and client.
- You know the following information:
  - The private key of the client
  - The static tunnel IP addresses and subnet masks of the client
  - The public key of the server
  - The static tunnel IP addresses and subnet masks of the server

Procedure

1. Open a terminal, and enter:

```
# nm-connection-editor
```

2. Add a new connection by clicking the **[+]** button.
3. Select the `WireGuard` connection type, and click **[Create]**.
4. Optional: Update the connection name.

5. Optional: On the `General` tab, select `Connect automatically with priority` .
6. On the `WireGuard` tab:
   i. Enter the name of the virtual interface, such as `wg0` , that NetworkManager should assign to the connection.
   ii. Enter client's private key.
   iii. Click **[Add]** to add peers:
      i. Enter the public key of the server.
      ii. Set the `Allowed IPs` field. For example, set it to:
         - The tunnel IP addresses of the server to allow only the server to communicate with this client.
         - `0.0.0.0/0;::/0;` to allow any remote IPv4 and IPv6 address to communicate with this client. Use this setting to route all traffic through the tunnel and use the WireGuard server as default gateway.

           Note that routing all traffic through the tunnel can impact the connectivity to other hosts based on the server routing and firewall configuration.
      iii. Enter the hostname or IP address and port of the WireGuard server into the `Endpoint` field. Use the following format: *`hostname_or_IP`*`:`*`port_number`*
      iv. Optional: If you use the client in a network with network address translation (NAT) or if a firewall closes the UDP connection after some time of inactivity, set a persistent keep alive interval in seconds. In this interval, the client sends a keep alive packet to the server.
      v. Click **[Apply]** .
7. On the `IPv4 Settings` tab:
   i. Select `Manual` in the `Method` list.
   ii. Click **[Add]** to enter the tunnel IPv4 address and the subnet mask.

iii. If you want to route all traffic through the tunnel, set the tunnel IPv4 address of the server in the `Gateway` field. Otherwise, leave the field empty.

Routing all IPv4 traffic through the tunnel requires that you included `0.0.0.0/0` in the `Allowed IPs` field on this client.

8. On the `IPv6 Settings` tab:

i. Select `Manual` in the `Method` list.

ii. Click **[Add]** to enter the tunnel IPv6 address and the subnet mask.

iii. If you want to route all traffic through the tunnel, set the tunnel IPv6 address of the server in the `Gateway` field. Otherwise, leave the field empty.

Routing all IPv4 traffic through the tunnel requires that you included `::/0` in the `Allowed IPs` field on this client.

9. Click **[Save]** to store the connection profile.

Verification

1. Ping the IP addresses of the server:

```
# ping 192.0.2.1
# ping6 2001:db8:1::1
```

2. Display the interface configuration of the `wg0` device:

```
# wg show wg0
interface: wg0
  public key: bnwfQcC8/g2i4vvEqcRUM2e6Hi3Nskk6G9t4r26nFVM=
  private key: (hidden)
  listening port: 51820

peer: UtjqCJ57DeAscYKRfp7cFGiQqdONRn69u249Fa4O6BE=
  endpoint: server.example.com:51820
  allowed ips: 192.0.2.1/32, 2001:db8:1::1/128
  latest handshake: 1 minute, 41 seconds ago
  transfer: 824 B received, 1.01 KiB sent
  persistent keepalive: every 20 seconds
```

To display the private key in the output, use the `WG_HIDE_KEYS=never wg show wg0` command.

Note that the output only has the `latest handshake` and `transfer` entries if you have already sent traffic through the VPN tunnel.

3. Display the IP configuration of the `wg0` device:

```
# ip address show wg0
10: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue
state UNKNOWN group default qlen 1000
    link/none
    inet 192.0.2.2/24 brd 192.0.2.255 scope global noprefixroute wg0
       valid_lft forever preferred_lft forever
    inet6 2001:db8:1::2/32 scope global noprefixroute
       valid_lft forever preferred_lft forever
    inet6 fe80::73d9:6f51:ea6f:863e/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

ADDITIONAL RESOURCES

The `wg(8)` man page

8.14. Configuring a WireGuard client by using the wg-quick service

You can configure a WireGuard client by creating a configuration file in the `/etc/wireguard/` directory. Use this method to configure the service independently from NetworkManager.

This procedure assumes the following settings:

- Client:
  - Private key: `aPUcp5vHz8yMLrzk8SsDyYnV33IhE/k20e52iKJFV0A=`
  - Tunnel IPv4 address: `192.0.2.2/24`
  - Tunnel IPv6 address: `2001:db8:1::2/32`
- Server:
  - Public key: `UtjqCJ57DeAscYKRfp7cFGiQqdONRn69u249Fa4O6BE=`
  - Tunnel IPv4 address: `192.0.2.1/24`
  - Tunnel IPv6 address: `2001:db8:1::1/32`

Prerequisites

- You have generated the public and private key for both the server and client.
- You know the following information:
  - The private key of the client
  - The static tunnel IP addresses and subnet masks of the client
  - The public key of the server
  - The static tunnel IP addresses and subnet masks of the server

Procedure

1. Install the `wireguard-tools` package:

```
# dnf install wireguard-tools
```

2. Create the `/etc/wireguard/wg0.conf` file with the following content:

```
[Interface]
Address = 192.0.2.2/24, 2001:db8:1::2/32
PrivateKey = aPUcp5vHz8yMLrzk8SsDyYnV33IhE/k20e52iKJFV0A=
*
[Peer]
PublicKey = UtjqCJ57DeAscYKRfp7cFGiQqdONRn69u249Fa406BE=
AllowedIPs = 192.0.2.1, 2001:db8:1::1
Endpoint = server.example.com:51820
PersistentKeepalive = 20
```

- The `[Interface]` section describes the WireGuard settings of the interface on the client:
  - `Address` : A comma-separated list of the client's tunnel IP addresses.
  - `PrivateKey` : The private key of the client.
- The `[Peer]` section describes the settings of the server:
  - `PublicKey` : The public key of the server.

  - `AllowedIPs` : The IP addresses that are allowed to send data to this client. For example, set the parameter to:
    - The tunnel IP addresses of the server to allow only the server to communicate with this client. The value in the example above configures this scenario.
    - `0.0.0.0/0, ::/0` to allow any remote IPv4 and IPv6 address to communicate with this client. Use this setting to route all traffic through the tunnel and use the WireGuard server as default gateway.
  - `Endpoint` : Sets the hostname or IP address and the port of the server. The client uses this information to establish the connection.
  - The optional `persistent-keepalive` parameter defines an interval in seconds in which WireGuard sends a keepalive packet to the server. Set this parameter if you use the client in a network with network address translation (NAT) or if a firewall closes the UDP connection after some time of inactivity.

3. Enable and start the WireGuard connection:

```
# systemctl enable --now wg-quick@wg0
```

The systemd instance name must match the name of the configuration file in the `/etc/wireguard/` directory without the `.conf` suffix. The service also uses this name for the virtual network interface.

Verification

1. Ping the IP addresses of the server:

```
# ping 192.0.2.1
# ping6 2001:db8:1::1
```

2. Display the interface configuration of the `wg0` device:

```
# wg show wg0
interface: wg0
  public key: bnwfQcC8/g2i4vvEqcRUM2e6Hi3Nskk6G9t4r26nFVM=
  private key: (hidden)
  listening port: 51820

peer: UtjqCJ57DeAscYKRfp7cFGiQqdONRn69u249Fa4O6BE=
  endpoint: server.example.com:51820
  allowed ips: 192.0.2.1/32, 2001:db8:1::1/128
  latest handshake: 1 minute, 41 seconds ago
  transfer: 824 B received, 1.01 KiB sent
  persistent keepalive: every 20 seconds
```

To display the private key in the output, use the `WG_HIDE_KEYS=never wg show wg0` command.

Note that the output contains only the `latest handshake` and `transfer` entries if you have already sent traffic through the VPN tunnel.

3. Display the IP configuration of the `wg0` device:

```
# ip address show wg0
10: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue
state UNKNOWN group default qlen 1000
    link/none
    inet 192.0.2.2/24 scope global wg0
       valid_lft forever preferred_lft forever
    inet6 2001:db8:1::2/32__ scope global
       valid_lft forever preferred_lft forever
```

ADDITIONAL RESOURCES

The `wg(8)` man page

The `wg-quick(8)` man page

PREVIOUS NEXT